Nayak, Richi (2008) *XML Data Mining: Process and Applications*, in Song, Min and Wu, Yi-Fang, Eds. *The Process and Applications of XML Data Mining*. Idea Group Inc. / IGI Global.

THE PROCESS AND APPLICATIONS OF XML DATA MINING

Richi Nayak

Faculty of Information Technology

Queensland University of Technology

S 835, Gardens Point, GPO Box 2434, Brisbane, QLD 4001, Australia

Ph: +61 7 3138 1976

Fax: +61 3138 1214

r.nayak@qut.edu.au

Keywords. Data mining, XML, XML structure mining, XML content mining

The Process and Applications of XML Data Mining

Richi Nayak
Faculty of Information technology
Queensland University of Technology, Brisbane, Australia

ABSTRACT
XML has gained popularity for information representation, exchange and retrieval. As XML material becomes more abundant, its heterogeneity and structural irregularity limit the knowledge that can be gained. The utilisation of data mining techniques becomes essential for improvement in XML document handling. This chapter presents the capabilities and benefits of data mining techniques in the XML domain, as well as, a conceptualization of the XML mining process. It also discusses the techniques that can be applied to XML document structure and/or content for knowledge discovery.

INTRODUCTION
The Web is an immense and dynamic collection of pages and services that includes countless hyperlinks, thus, it provides a rich and diversified data mining source. Currently, the majority of this information is in Hyper Text Markup Language (HTML). HTML tags are primarily formatting markup and were designed to convey technical reports. Some internal structural information can be inferred from them, (e.g. <h1> indicating important information) but they hold no semantic information regarding content. With an increasingly distributed corporate world and progression to Web 2.0, HTML is considered an inferior means of data exchange.
To overcome these limitations, XML (eXtensible Markup Language) uses custom-defined tags to describe the data and the structural relationships of data within a document. XML is a subset of SGML (ISO 8879) and is defined by the World Wide Web Consortium (W3C) (Yergeau et al., 2004). XML tags describe the structural and semantic meaning of information in text documents thus make the XML documents semi-structured and self-describing. XML is rapidly becoming the standard for exchanging and representing data. Many information sources have already or are beginning to structure their external view as a repository of XML documents, regardless of their internal storage mechanism.
As XML data becomes more abundant, the ability to gain knowledge from XML sources decreases due to their heterogeneity and structural irregularity. Several advanced data processing techniques are required to retrieve and analyse such large amounts of semi-structured data. Automatic storage of XML documents in the form of relational or object-oriented data has been actively studied by database researchers (Abiteboul et al., 2000) (Lee et al., 2002). Other researchers have successfully stored XML documents in native XML databases (Pardede, 2006). Consequently, several query languages for various XML data sources have been developed (Boag et al.). The use of these query languages is limited, for example, users need to know what kind of information is to be accessed and only limited inputs and outputs are acceptable. Additionally, indexing based on structural similarity and/or based on groupings of XML documents sharing frequent sub-structures are needed to support effective document storage and retrieval (Nayak et al., 2002).
Data mining techniques such as clustering (Jain et al., 1999) can improve XML document storage and retrieval by grouping XML documents according to their structural similarity. Computation of structural similarity is also a great value in managing the Web data. Many techniques of extraction and integration of relevant information from the Web data sources require grouping the Web data sources according to their structural similarity (Flesca et al.,

2005). Moreover data mining (Fayyad, 1995) techniques allow the user to search for unknown facts, information that is hidden behind the data, and also allow users to pose more complex queries. For example, after identifying similarities among various XML documents using clustering, links between tags within a group of XML documents can be analysed using association mining. This may prove useful in analysis of e-commerce web documents and subsequently in personalisation of web pages.

There is a considerable body of research on mining useful information from numerical, symbolic and text data (Han & Kamber, 2001). There have been some progress on using XML as a language in data mining process models such as (1) Predictive Model Markup Language (PMML) (Wettschereck, 2001) for utilizing XML to specify several kinds of data mining models, (2) XML based Data Mining Specification Language (XDMSL) for describing the data mining process (Kotásek & Zendulka, 2002) and (3) Log Markup Language for utilizing XML to structurally express the contents of Web server log files(Punin et al., 2001).

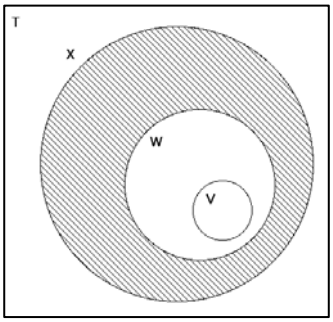
Research on developing data mining techniques for XML documents is gaining momentum (Nayak & Zaki, 2006). The characteristic of XML that adds semantic and structural aspects to document contents offers new data mining opportunities. At the same time, this also makes the data mining process challenging by including the semantic and structural aspects into analysis.

Given the irony that humans produce far more data than they can ever analyse, the development of XML mining techniques must keep pace with the development and implementation of XML technology itself. This chapter is motivated by the potential of these two mutually beneficial technologies. It first briefly describes the XML data and the equivalent tree representation. It then presents a classification of XML mining methods, a discussion of mining applications such as classification, clustering and association followed by a summary of tools and techniques that can be successfully applied to the content or structure of XML documents for knowledge discovery. This chapter provides an up-to-date survey of XML mining and will include both academic efforts and commercial offerings.

REPRESENTATION OF XML DOCUMENTS

This section provides background information on XML. Let all textual Web objects be the set T. Let web pages containing XML - to be called XML data - be X, such that $X \subseteq T$. There are two types of XML data: XML documents and XML schemas. A XML schema provides the data definitions and structure of the XML document (Abiteboul et al., 2000). XML documents are the instances of a schema**,** a snapshot of what the document may contain. A schema includes allowable elements and attributes and the number of occurrences of elements and other constraints. A schema for a document may be included as both internally and externally (within the same file or in a different file, respectively).

In a heterogeneous and flexible environment such as the Web, it cannot be assumed that each XML document has a schema defining its structure. Additionally even if such exists, it may have undergone multiple modifications. Consequently, all XML or Web data cannot be automatically classed as XML documents. Strictly, web data or XML data are classed as XML documents only if they are well-formed. To be well-formed, a page's XML must have properly nested tags, unique attributes (per element), one or more elements and only one root element, as well as a number of schema-related constraints. Well-formed documents have a schema but may not conform to it. Valid XML documents are a subset of well-formed XML documents. A valid XML document must additionally conform (at least) to an explicitly associated schema. Figure 1 depicts the various types of XML data and how they are related.



T: textual web data,
X: XML data,
▨: ill-formed XML data,
W: well-formed XML documents,
V: valid XML documents

Figure 1: Relationship between various XML data

```
<?xml version=”1.0” encoding=”UTF-8”?>
<BookStore>                                      <!DOCTYPE BookStore [
    <Book>                                       <!ELEMENT BookStore (Book+)>
      <Title> Introduction of XML </Title>       <!ELEMENT Book (Title, (Author)*,
      <Author>                                                 ISBN, Publisher)>
            <fName> Smith </fName>               <!ELEMENT Title (#PCDATA)>
            <lName> Andrew </lName>              <!ELEMENT Author(fName,mName?,lName)>
      </Author>                                  <!ELEMENT ISBN (#PCDATA)>
      <ISBN> 2564.6554.5545 </ISBN>              <!ELEMENT Publisher (#PCDATA)>
      <Publisher> McGraw-Hills </Publisher>      ]>
    </Book>
      .
      .
</BookStore>
```

Figure 2: Example of a XML document and its respective DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema,
    targetNamespace=http://www.books.org,xmlns=http://www.books.org,
            elementFormDefault="qualified">
  <xsd:element name="BookStore">
     <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="Book" minOccurs="1" maxOccurs= "unbounded"/>
        </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
    <xsd:element name="Book">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="Title" minOccurs="1" maxOccurs="1"/>
    <xsd:element ref="Author" minOccurs="1" maxOccurs="unbounded"/>
                <xsd:element ref="ISBN" minOccurs="1" maxOccurs="1"/>
                <xsd:element ref="Publisher" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="Title" type="xsd:string"/>
    <xsd:element name="Author" type="xsd:string"/>
    <xsd:element name="Date" type="xsd:string"/>
    <xsd:element name="ISBN" type="xsd:string"/>
    <xsd:element name="Publisher" type="xsd:string"/>
</xsd:schema>
```

Figure 3: Example of the respective XSD For the above document

There are several XML schema languages that allow the structure of XML documents to be described and their contents to be constrained[1] . Only two are commonly used, namely DTD (Document Type Definition) and XML Schema or XML Schema Definition (XSD). The DTD language is considered limited as it only supports a limited set of data types, has loose structure constraints and limits content to textual. To overcome the above limitations of DTD, XSD provides features, such as simple and complex types, rich datatype sets, occurrence constraints and inheritance. An XML schema is usually comprised of a set of schema components, such as type definitions and element declarations. They can be used to assess the validity of well-formed elements. It is believed that XSD with its flexibility will soon more popular than DTD[2] . Throughout this chapter, the term 'schema' is used to express both XML-DTD and XML-Schema unless specified. The term 'XML data' is used to express both XML documents and XML schemas. Figure 2 illustrates an XML document and its corresponding DTD. Figure 3 shows its respective XML Schema.

XML MINING: TAXONOMY

For several years data mining (DM) has been used to extract meaningful knowledge from large amounts of data. Mining of XML documents differs significantly from that of numerical, symbolic and text data. XML mining is the use of DM techniques to automatically discover and extract information from sources of XML documents. The fact that data is represented in hierarchical format in XML documents poses a challenge for DM. Moreover, XML documents can be designed with many flexibilities and minimal restrictions. Many see this as one of the greatest strength of XML, however, this makes the process of document handling difficult.

Consider parts of two documents: *<craft>boat building</craft>* and *<craft> boat </craft>*. The intended interpretation of the former is 'occupation', and of the latter 'vessel'. Similarity of the content does not distinguish the semantic intention of the tags. These two fragments will be found to be very similar based on words common to the two sets {craft, boat, building, craft} and {craft, boat, craft}. Use of structure mining in this case provides the probability of a tag's having a particular meaning. For example, a mining rule inferred from a collection of XML documents is "*80% of the time, if an XML document contains a <craft> tag then it also contains a <driver> tag*". Such a rule now helps determine the appropriate interpretation for such homographic tags. Hence, mining for the structure and content of documents can clarify when two similar documents are actually completely different, given homograph tags.

There are many benefits and applications that can be obtained with the utilisation of XML data mining techniques such as:

- Enhancing information sharing among various industries and government by proposing techniques for organizing and integrating various heterogeneous and distributed XML documents.
- Improving the accuracy and speed of the XML-based search engines in retrieving the relevant portions of data (1) by suggesting XML documents according to the similarity of their structure and content, and (2) by discovering the links between XML tags that occur together within the XML documents. For example, a DM rule can discover that "*<telephone> tags must appear within <customer> tags*" from a collection of XML documents. This information can be used by searching only <customer> tags when executing a query about finding <telephone> details thus making the information retrieval efficient.

[1] XML Schema: http://www.w3.org/XML/Schema

[2] Introduction to XML Schema by Refsnes Data: http://www.w3schools.com/schema/schema_intro.asp

- Improving XML document handling and achieving efficient searches on relevant documents by using the developed set of predefined document classifications.
- Better representation of information provided in Web sites with better restructuring by recommending (1) Web links that occur together; and (2) Web documents that are similar in structure and content.

Mining of XML documents differs significantly from other structured data. XML mining includes mining of structures as well as content from XML documents (Nayak et al., 2002), depicted in figure 4. Mining *XML content* is generally carried out in the context of known XML structure, possibly determined by *XML structure mining*. Content mining may, however, also play a role in clarifying XML structure. Therefore to avoid information loss, the structural and contextual data are frequently combined for the best use of XML documents.

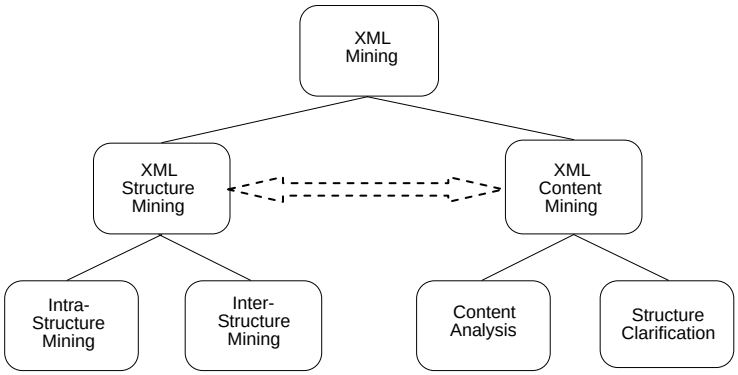


Figure 4: A taxonomy of XML Mining

Next both structure and content mining is discussed with the application of data mining operations such as classification, clustering and association, and the type of XML material available for input to these procedures. Technical details of measurements such as criteria for classification or similarity metrics for clustering will not be covered in this section, since the main objective is to establish usage and benefit of data mining in XML.

XML Structure Mining

XML is semi-structured data thus mining for XML structure provides insights. Element tags and their nesting therein dictate the *structure* of an XML document (Yergeau et al., 2004). For example, the textual structure enclosed by <*author*>… </*author*> is used to describe the "author" tuple and its corresponding text in the document. Tags in XML are user-defined and describe the area of interest. For example, <manufacturer>, <model>, and <colour> tags can be used to describe the car information for the automobile industry. Since XML provides a mechanism for tagging names with data (to describe the data), retrieval of more accurate information on XML documents structure can be facilitated with the use of data mining. XML structure mining is essentially mining for schema including *intra-structure mining*, and *inter-structure mining*.

Intra-structure Mining

Intra-structure mining is concerned with the structure within an XML document(s). Knowledge is discovered about the internal structure of XML documents.

The *classification* task of data mining can be applied to map a new XML document to a predefined class of documents. XML document structure can be read directly or via the document's schema. A document schema provides a definitive description of a document, while a document instance only shows the content of the document. Because the document

definition outlined in a schema holds true for all document instances of that schema, the result produced from classifying schemas would also hold true for all document instances of the classified schemas and can be reused for any other instances of these schemas.
A schema is interpreted as a description of a class of XML documents. Let us assume that each document is accompanied by a schema. In the absence of a schema, a XML document is parsed and the structure is extracted and modelled. Given a collection of schemas as a training set, the objective of this task is to classify new XML schemas according to this training set of schemas. Both the semantic and structural similarities are considered in classifying a schema into a class. This task is most easily performed on valid XML documents. With schemas already defined for the new XML document, the classification task can proceed by comparing the classification schemas with the new schema. For any XML document with an associated schema, it should first be validated. It is important to distinguish between valid XML and well-formed XML with incorrectly associated schema. For well-formed XML, an attempt is made to parse the documents according to the classification schema. A successfully parsed document is classified as an instance of the relevant schema.
Ill-formed XML with associated schemas may also be classified if enough of the document is parsed before an error occurs. Then the classification could be used to 'rescue' any potentially valuable information. The task will be most difficult (but still possible) for XML with no associated schemas. In this case, the similarity will be found between the classification schemas (classes) and the document structure.
The *clustering* task of data mining can be used to identify similarities among XML documents. The structure of each XML document is inferred and modelled as a labelled tree. Each node in the tree has information about that element, e,g, name, cardinality, position etc. A clustering algorithm takes a collection of trees and groups them on the basis of semantic and structural similarity. These similarities are then used to generate new schema. As a generalisation, the new schema becomes a superclass to the training set of schemas. This generated set of clustered schemas now can be used in classifying new schemas. The superclass schema can also be used in integration of heterogeneous XML documents for each application domain. This allows users to find, collect, filter, and manage information sources on the Internet more effectively.
The *association rules* discovery task of data mining can describe relationships between tags which occur together in XML documents. A XML document/schema can be represented as a tree structure. Each tree branch (or path) is considered a transaction. By transforming the tree structure of XML into pseudo-transactions, it becomes possible to generate rules of the form "if an XML document contains a <craft> tag then 80% of the time it will also contain a <driver> tag." Such a rule is then applied in determining the appropriate interpretation for homographic tags (wherein words which are like one another in form have distinctly different meanings).

Inter-structure Mining

Inter-structure mining is concerned with the structure between XML documents. Knowledge about the relationship between subjects, organizations and nodes on the Web is discovered.
Clustering schemas involve identifying similar schemas according to the linguistic and hierarchical closeness. The clusters are used in defining hierarchies of schemas. The schema hierarchy overlaps instances on the web, thus discovering authorities and hubs (Garofalakis, 1999). Creators of schemas are identified as authorities, and creators of instances are hubs. Additional mining techniques are required to identify all instances of schemas present on the web. The following application of classification can identify the most likely places to mine for instances. *Classification* is applied with namespaces and URIs (Uniform Resource

Identifiers). Having previously associated a set of schemas with a particular namespace or URI, this information is used to classify new XML documents linked with this URI.

XML Content Mining

Content is the text between each start and end tag (Yergeau et al., 2004) in XML documents. Mining for XML content is essentially mining for values (an instance of a relation). The semi-structured nature of XML poses a challenge for content mining. XML content mining can further be divided into two tasks: *content analysis* and *structural clarification.*

Content Analysis

Data mining of flat text files has been successfully conducted as the content of the text files is treated as a bag of words or terms. Tasks similar to those performed on other text documents can be performed on XML documents. However, XML represents its data in a hierarchical structural format that makes content analysis harder than it is for plain text. One has to consider the granularity and the need for indexing at various levels of abstraction (e.g., whole XML documents vs. parts of XML document) in mining.

Classification is performed on XML content, labelling new XML content as belonging to a predefined class. A massive search would be required to match the contents of a new XML document with every document in the collection.. To reduce the number of comparisons, firstly, the schema of a new document is classified by a pre-existing schema. Then, only the instance classifications of the matching schema need to be considered in classifying a new document.

Clustering on XML content identifies the potential for new classifications. Consideration of schemas leads to a fast clustering process: similar schemas are likely to have a number of value sets. For example, all schemas concerning vehicles will have a set of values representing cars, another set representing boats, *etc*. However, schemas that appear dissimilar may have similar content. Mining XML content inherits some problems faced in text mining and analysis. Synonymy and polysemy can cause difficulties, but the tags surrounding the content can usually resolve ambiguities.

Structure Clarification

Content provides support for alternate clustering of similar schemas. Content may prove important in clustering schemas that appear different but have instances with similar content. Due to heterogeneity, the occurrences of synonyms increase. Mining these schemas provides information such as: Are separate schemas actually describing the same thing, only with different terms? While thesauruses are vital, it is impossible for them to be exhaustive in the English language, let alone be so in *all* languages. Vice versa, schemas provide support for alternate clustering of content. Two XML documents with distinct content may be clustered together given that their schemas are similar. Schemas appearing similar are actually completely different, given homographs. For example, consider: <craft>boat building</craft> and <craft>boat</craft>. Interpretation of the former is occupation and of the later vessel. The similarity of content does not distinguish the sematic intention of tags. Mining in this case provides probabilities of a tag's having a particular meaning, or a relationship between meaning and a document.

XML MINING: PROCESS

The XML mining process combines the pre-processing, pattern discovery and post-processing. Pre-processing the XML data infers relevant XML structures and contents from the specific resources. For pattern discovery, application of classification, clustering and

association mining techniques to pre-processed data identifies interesting information. Lastly, the mined patterns are validated and interpreted in the post-processing phase.

Pre-processing: Inferring XML Structure

The main goal of pre-processing is to successfully infer structures from XML documents, so a DM technique can identify interesting patterns. The output of this process is mostly a tree or a graph representation that yields the structure of the document or schema. It is not mandatory for an XML document to have a schema that defines its data and structure. A schema describes the grammar of an XML document and allows the document to be parsed. XML documents are classified as "ill-formed, "well-formed" or "valid" according to their structure. Based on this classification, there are two cases of inferring structures: one is from well-formed or valid documents and another is from ill-formed XML documents.

Inferring structure from Well-Formed or Valid XML Documents

Given the schemas attached to the well-formed or valid documents, the structure of these documents can be easily inferred by traversing the document. The inferred structure can be represented in tree format, or a relational representation of the data can be created. The structure can be presented as a table with relational attributes to contain the embedded data. If the hierarchy of the attributes is deeper then database techniques such as the addition of more relations and foreign keys and/or normalization methods can be used to accommodate the structure and the data. The structure can be inferred most easily from valid XML documents. For a well-formed XML document, it is necessary to check the validity of the document with respect to its associated schema, in case an inappropriate schema is defined. A variety of XML tools, known as validating parsers, have been developed to verify the conformity of well-formed XML documents with their schemas. Moreover, the well-formed documents may not always have an accompanying schema since the presence of a schema is not mandatory. Schema extraction tools are able to generate schemas from the semantic structure of these documents

DTD Generator is a commonly used tool to generate the DTD for a given XML document (Kay, 2000). It identifies a DTD for every XML document hence a separate set of rules for each XML document in a collection of documents is defined rather than an overall set of rules for the collection. Tools such as XTRACT (Garofalakis, 2000) and DTD-Miner (Moh, 2000) infer an accurate and semantically meaningful DTD schema for a given collection of XML documents. These tools require a relatively homogeneous collection of XML documents. In such heterogeneous and flexible environment as the Web, it is unreasonable to assume that XML documents related to the same topic have similar document structure.

Due to limitations in using DTDs as an internal structure, many researchers propose the extraction of XSD (Feng, 2002) (Vianu, 2001). XSD is also not obligatory in XML documents hence extraction of structure information from XML documents is necessary to create the XML Schema. A XML schema extraction algorithm based on the Extended Context-Free Grammar**s** (ECFG) with a range of regular expressions is proposed (Nestorov, 1999). A semantic network-based design is also presented to convey the semantics carried by the XML hierarchical data structures and to transform the model into an XML Schema thus increasing user understanding of the documents' semantic structure and content as well as the relationships within them (Feng, 2002).

Inferring structure from Ill-Formed XML Documents

In practice, XML documents often have no schema, and no fixed or rigid structure. Schema for such *ill-formed* XML documents can be inferred by applying the structure extraction approaches developed for semi-structured documents but not all techniques can effectively

infer the structure required by further DM algorithms. They do not include the necessary granularity, the various levels of abstractions and the nesting of tags For instance, the NoDoSe tool (Adelberg, 1998) is primarily used for determining the structure of semi-structured documents, and it does not support hierarchy as in XML. The extraction algorithms proposed by (Myaeng, 1998) consider both structure and contents in semi-structured documents, however, their purpose is to query and build an index. They are difficult to use and must be altered and adapted prior to the application of data mining algorithms.
For extraction of structures from an ill-formed XML document, the Object Exchange Model (OEM) data and its corresponding data graph can produce the most specific (accurate and concise) data guide/schema (Nestorov, 1999) (Wang, 2000) (Nayak et al., 2002). The TreeSketch and XSketch methods facilitate query processing by extracting structural summaries (Polyzotis et al., 2004). In summary, these methods rely on a generic graph-summarization model, which captures the basic structure of XML documents, augmented with appropriate distribution information at different levels of granularity. Such methods are more applicable than DTD/XSD since most XML documents have no schema and may not conform to it if they do. Some semi-structured data are the result of queries. In such cases it is possible to derive the structure from the query that generated the data and doing so is a better choice than extracting the schema from the data.

Pre-processing: Inferring XML Content
To discover knowledge in XML documents, it is necessary to query XML tags and content and several query languages, either designed specifically for XML or those used for semi-structured data in general are available.

Query Languages for Semi-structured Data
XML represents a subset of semi-structured data. Semi-structured data is described by the grammar of ssd-expressions (semi-structured data). The translation of XML to an ssd-expression is easily automated (Abiteboul et al., 2000). Figure 5 shows an XML description of a person object and an equivalent ssd-expression. Query languages for semi-structured data exploit path expressions. In this way, data can be queried to a variable depth. Path expressions are elementary queries that return the results as a set of nodes. However, results must be returned as semi-structured data and path expressions alone cannot do this. Combining path expressions with SQL-style syntax provides greater flexibility in testing for equality, performing joins, and specifying the form of query results. Two such languages are Lightweight Object Repository (Lorel) (Abiteboul, 1997) and Unstructured Query Language (UnQL) (Fernandez, 2000). Lorel took an object-oriented approach and minimized dependence on predetermined schema information. UnQL relies more on path expressions and requires greater precision. Figure 6 shows a query both in Lorel and UnQL and, as well, it specifies the name of a new node and performs an equality test on the name.

XML: <person> <name> Kym</name> <age> 25 </age> <person>
ssd-expression: { person : { name : “Kym”, age : 25 } }

Figure 5: An example of XML and ssd-expression

Lorel: Select newNode: X From person.age X Where person.name = “Kym”	UnQL: Select newNode: X Where { person: {name: Y, age: X} } in db, Y = “Kym”
Result: { newNode: 25 }	

Figure 6: A Query written in Lorel and UnQL and its corresponding Result

<table>
<tr><td>XML-QL Query:
Where <person>
<name>Kym</name>
<age> $a </age>
<person> in db
Construct <newNode>
<age> $a </age>
</newNode></td><td>XSL Quey:
<xsl:for-each select = “person [name =“Kym”]”>
<newNode> <age>
<xsl:value-of select=”age”/>
</age> </newNode>
</xsl:for-each></td></tr>
<tr><td colspan="2">XQuery query:
for $b in doc("db.xml") /db/person
where $b/name = “Kym”
return <newNode>
<age> $b/age </age>
</newNode></td></tr>
<tr><td colspan="2">Result: <newNode>
<age> 25 </age>
</newNode></td></tr>
</table>

Figure 7: A Query written in XML-QL, XSL and XQuery

Query Languages for XML

XML-QL, XSLT, XML-GL, YATL and XQuery are designed specifically for querying XML. XML-QL (Garofalakis, 2000) combines regular path expressions, SQL-style query techniques and XML syntax. Extensible Stylesheet Language Transformation (XSLT) is not implemented as a query language, but is akin to a query in its transformation of XML to HTML and its ‘select pattern’ mechanism for information retrieval. XML-GL (Ceri, 1999) is a graphical language for querying and restructuring XML documents. YATL is intended to capture a large and useful class of data transformation for querying multiple XML data sources. YATL brings together information from multiple data sources in one query. XQuery (Boag et al.) uses the structure of XML to express queries across several data types, whether physically stored in XML or viewed as XML via middleware. XQuery operates on the abstract, logical structure of an XML document, rather than its surface syntax. These queries produce the output in XML, thus, allow the transformation of XML data from one schema to another.

Pattern Discovery: Combining structure and content

Many XML data mining techniques mine useful information from the structure and content of XML. The techniques can be divided into three areas: clustering, classification and association.

XML Clustering

There have been a myriad of techniques developed for finding similarity between documents or schemas. These techniques are used mainly in data/schema integration or query approximation. As well, these techniques facilitate the clustering process. They do by considering the XML semantic information (linguistic and context elements) as well as the hierarchical structure.

The process usually starts by representing the XML document or schema into a tree presentation. Semantic similarity measures use acronyms, synonyms, hyponyms, hypernyms of names used to compare corresponding elements in each of the trees and, as well, they consider the hierarchical positions of elements in the tree. Sequential pattern mining algorithms (Agrawal & Srikant, 1996) have been used by many researchers (Nayak, 2007) (Lee & Park, 2004; Leung et al., 2005) to measure structural similarity. These algorithms

represent a tree by a set of paths/sequences. A path is represented by a unique sequence of element nodes following the containment links from root to leaf nodes. The sequential pattern algorithm computes the maximal similar paths between XML documents. The combination of semantic and structural similarity is represented as a similarity matrix. K-means or hierarchical agglomerative clustering algorithms (Jain et al., 1999) generate clusters of XML documents.

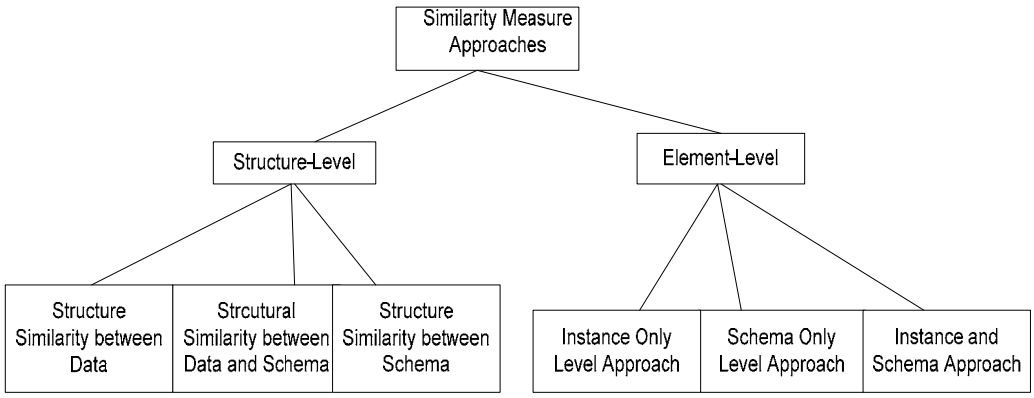


Figure 8: A Classification of Similarity Measure Approaches

A classification of these approaches is presented in figure 8. The ***structure-level similarity approaches*** *detect and measure three different sets of data*; (1) structural and content similarities between documents (Dalamagas et al., 2004; Flesca et al., 2005; Huang, 1997; Lee et al., 2002; Nayak & Xu, 2006), (2) the structural similarity between documents and schemas (Bertino et al., 2004), and (3) the structural and content similarity between schemas (Nayak, 2007; Nayak & Xia, 2004).

The approaches using data from the first and third alternative rely on the notion of "tree edit distance" developed in combinatorial pattern matching (Zhang & Shasha, 1989). The problem is to compute the minimum distance between two trees *T1* and *T2 and th*ere are three common editing operations available: *changing*, *deleting*, and *inserting* a node. For each of these operations a cost is assigned and it depends on the labels of the nodes involved. The problem is to find a sequence of such operations (an *edit script*) transforming *T1* into *T2* with minimum cost. The distance between *T1* and *T2* is then defined to be the cost of such a sequence.

The use of the second set of data relies on measuring the structural similarity between data and schema in the context of XML. Some of these techniques present documents as edge-labelled graphs, ignoring the constraints on the repeatability, or as element alternatives in XML schemas. Additionally, some techniques cannot be directly applied to cluster documents without knowledge of their schemas, and dissimilarities among documents referring to the same schema cannot be identified. However, these approaches take into account the context of element that strongly contributes to determine which information that element models.

The ***element-level similarity matching*** approaches known as *schema matching* determine the semantic correspondence between elements of two schemas. These methods use the document schema to cluster XML documents. Relevant schema information is used to efficiently determine the similarity of corresponding elements in XML documents. . The document schema provides a definitive description of the document, while document instances represent examples of content. The document definition outlined in a schema holds true for all document instances of that schema, hence schema clustering results hold true for all document instances and can be reused for other instances.

The main difference between element-level matching approach and structure-level matching approach is that in the former, similarity determination is based primarily on elements of the trees, in particular, their semantic names and name structures similarity. On the other hand, structure-level matching determines whole tree structure similarity and ignores detailed elements in the tree. The tree edit problem treats the label of each node in the tree as a second preference. For instance, the cost of relabelling is assumed to be less computationally expensive than that of deleting a node with the old label and inserting a node with the new label. Thus schema matching uses internal tree elements, whereas the tree edit distance approach matches tress at a higher level. The tree-edit distance approach addresses only the existence of different elements in two trees not their cardinality.

Researchers have approached schema matching for XML data at three different levels as shown in figure 8. *Instance-based matchers* use either meta-data and statistical data collected from data instances to annotate the schema or directly correlated schema elements (Kurgan et al., 2002). *Instance only level approaches* sometimes fail to capture the structure information of the XML data. Machine learning techniques are used to improve accuracy but can be very computationally expensive.

Schema-based matchers consider only schema information, not instance data. Schema information includes tag names, descriptions, relationships, constraints, etc. Schema matching at *schema only level approaches* can be used for mapping a collection of heterogeneous XML-Schemas (Do & Rahm, 2002; Jeong & Hsu, 2001; Lee et al., 2002; Madhavan et al., 2001; Melnik et al., 2002; Nayak, 2007; Nayak & Xia, 2004). However, the absence of instance data can result in increased element mismatch. Therefore the accuracy of the mapping recommended by the schema only level approaches depend on the technique used for linguistic and structure matching. The instance only and schema only level approaches have difficulty finding similar elements between XML documents. Therefore many researchers have combined both the instance and schema information for schema matching (Doan et al., 2001). The instance and schema approaches however need both the XML documents and their associated schema definitions to be available for the mapping.

XML Association Mining

XML sources are generally represented as an ordered-labelled or unordered-labelled tree. The task is to build up associations among trees (including sub-trees, substructures, sub-graphs and paths) rather than items as in traditional mining. The frequent substructure (tree) mining extracts substructures (sub-trees, sub-graphs or paths) which occur frequently among a set of XML documents or within an individual XML document. These frequent substructures generate association rules. However, the frequent substructures are hierarchical and counting support requires more than just the joining of flat sets.

Generation of Frequent substructures: Let CS = {[C]1, [C]2,.. [C]d} be a set of initial *candidate* substructures sets, where *d* is the depth of the tree. This is different from traditional association mining (AM) in which there is no predefined candidate set, instead one is generated incrementally by merging elements in the frequent set of the previous round. In this hierarchical structure, a candidate set (CS) already exists. Additionally, in each round, the merging of current *candidate* sets derives a larger frequent fragment set. The search space for finding frequent structures is much larger than that for traditional association mining data sets thus it requires more effective pruning strategies (to eliminate the candidate item-sets in previous rounds) and merging strategies (to combine candidate item-sets in next round). Researchers have also utilised the mining of closed frequent trees to reduce the number of generated patterns (Kutty, 2007).

Recently a number of researchers have developed algorithms able to detect frequently occurring substructures from structural data collections. These include AGM, FSG, TreeMiner and gSpan (Paik et al., 2005; Zaki, 2002) . (Chi et al., 2005) gives a good overview of the frequent tree mining. An issue to consider with these algorithms is that they account for the dynamic nature of the XML data. To overcome this, (Zhao, 2007) have developed a frequently-changing structures mining technique that considers the changing nature of XML data. It aims to extract structures that change frequently from the sequence of historical XML versions. The structure which refers to "inserts" and "deletes" and the content which refers to "updates" of XML documents can change frequently. It is important to understand such changes in different versions of the same document**.**
Many XML DM techniques employ frequent sets in the process of classification of XML data as well as in the process of clustering and association rule generation.

Generation of Association Rules: A number of techniques use the expressive power of the query languages to extract association rules (Braga et al., 2002), or rely on the traditional framework with an XML interface (Edmonds, 2005; Kotasek, 2000). This requires user familiarity with the internal structure and content of the documents(s). Examples of user input include the XPath expression selecting the parent nodes of the data items to be mined and XPath expressions relative to that node locating the output and input values (Edmonds, 2005).
The XMINE rule operator extract association rules from XML documents using the SQL-like format (Braga et al., 2002). However XML data must be mapped to a relational structure before performing association. This requires powerful pre-processing, and may result in information loss during conversion. (Wan, 2004) used XQuery expressions to extract association rules from XML data and calculate *support* and *confidence*. This technique is limited that it fails to account for the structure of the XML data. For more complex XML data, transformation may be required before applying the XQuery expressions.
XAR-Miner transforms a small XML document into an indexed XML Tree (IX-tree–bi-directional linking between parent and child nodes) and transforms **a** large XML document into multi-relational databases (Zhang, 2004). XPaths for each relational database are created during data transformation maintaining the hierarchical information in the original XML document. A set of paths between the instances of related concepts are extracted from either the IX-tree or relational database for association rules mining. These paths (known as meta-pattern) are then generalized, eliminating any unnecessary meta-patterns to maximize the significance of the association rules. Based on this meta-pattern, XAR-Miner generalizes the raw XML data and generates association rules based on the user need using the Apriori algorithm.
The generation of association rules from the frequent hierarchal trees remains an unsolved problem.

XML Classification Mining
The classification task is applicable to a wide variety of problems in XML, however, it has not been studied well. Classification of XML documents requires the identification of structural rules. In the training phase, a set of structural classification rules are built and can be used in the learning phase to classify data of unknown class. The efficiency of existing XML document classification algorithms is limited by their inability to explore the structural information. A few researchers have developed generic classifiers (e.g., information retrieval (IR) based and association based) as well as specific classifiers (e.g. rule based according to structures) for XML.
The IR-based methods treat each document as a "bag of words". These methods use the actual text of the XML data but not the structural information inside the documents. The association-

based methods use the associations among different nodes visited in a session in order to perform the classification. An effective rule-based classifier for XML is XRules (Zaki & Aggarwal, 2003) , a method that uses a set of structural rules for XML document.classification.
XMiner (Zaki, 2002) uses frequent sub-trees in a collection of XML trees to mine a set of rules. In the training phase, it produces a set of structural classification rules that can be used in the learning phase to classify data of unknown class. XRules has shown to provide better XML classifiers in comparison to both the IR and association based classifiers.
(Theobald, 2003) explores the structure, annotation and ontological knowledge from XML data to facilitate automatic classification of XML data. It uses the support vector machine (SVM) technique in the training phase in which a set of tags (element name) and text terms are used. This technique computes separating lines (known as hyperplanes) between feature space objects from different classes. These separating lines can be used to test unseen data in the learning phase. This technique is based on the assumption that the tags are more important than text terms in exploiting the structural and ontological information from XML documents.
(Edmonds, 2005) uses the traditional framework with an XML interface to pre-process the data for training. It performs a statistical analysis of the pre-processed data and then creates a fuzzy decision tree before converting the result into Metarule format. The mapping of the XML data into a relational structure may result in information loss, and also requires an additional processing.

Post-processing: Interpreting mined patterns
Post-processing for the discovery of useful knowledge involves the analysis and assimilation of the generated XML pattern models. Due to the variety of tool-specific parameters, the resultant model and its performance must be properly interpreted. The mining model should be visualized in user-friendly fashion. As well, the generated prediction model should be able to classify unseen values using the user's tool. Extensive ongoing research into the post-processing phase of XML mining aims to improve the usability of data models. The following section identifies the evolution of interpretation approaches.

Conventional Approaches
In conventional approaches data models generated from a mining algorithm are treated differently depending on the application or mining tool being used. Such tools include OLAP (OnLine Analytical Processing), Relational DB and other data mining specific tools. With the use of these tools, problems occur when complex XML mining implementations are related to different/ XML-enabled databases and different application vendors, such as IBM, Oracle or Microsoft. Each tool has its own post-processing module that it uses to communicate the obtained result. In traditional mining techniques, this limitation exists regardless of the documents or area mined. In other words, there is difficulty in sharing data models obtained from multiple sources. It is necessary to deal with differences between applications and tools in order to share patterns generated from the mining process. However recent developments can output XML patterns in format which allows simpler and more flexible data mining applications.

Current Approaches
Recently**,** XML based markup languages that describe the data mining process are employed as part of the data mining post-processing. Discovered patterns can thus be interchanged among conforming data mining and analytical applications. The integrated data mining tools have tremendous potential for expanding the interoperability of the XML documents.

Recent developments include (1) Predictive Model Markup Language (PMML) (Wettschereck, 2001) which uses XML to specify several kinds of data mining models and (2) Log Markup Language which uses XML to structurally express the contents of Web server log files (Punin et al., 2001). These facilitate integration and analysis of the data collected from various web server log files and allow a better understanding of theuser's web site.
PMML (Predictive Modeling Markup Language), introduced by Data Mining Group (DGM), describes the structure and content of data mining models in the format of XML. A set of DTDs included in PMML is used to support several types of data mining models (Wettschereck, 2001). After a discovered XML pattern model is generated by a data mining algorithm, it is stored in the PMML format and thus allows model interchange. By implementing PMML, XML documents from multiple sources can be mined without consideration of differences between those sources and various applications used.
XDMSL (XML Data Mining Specification Language) extends the markup language approach to the whole process of knowledge discovery, including the source data model, data transformations, prior domain knowledge, data mining task description and knowledge discovered from data mining task (Kotásek & Zendulka, 2002). Many applications have not standardized the approach of XDMSL. To address this issue, XDMQL (XML Data Mining Query Language) is likely to be used for data exchange between different data mining system components as a part of the XDMSL implementation.
The above two languages are platform-independent, extensible and robust and are thus able to support information exchange in heterogeneous and modular environments.

COMMERCIAL USE OF XML IN DATA MINING

One of XML major advantages is its ability to manage the variety of data sources, types and structures that businesses transfer over the Internet. Despite some differences between the XML data and the typical historical relational data associated with data mining, there is a driving force in using the Internet as a medium for analytical data. XML itself is effective in transmitting and sharing data over the Internet. Companies want to extend this advantage into analytical data as well. Using XML data in the mining process is quite an innovation and is made possible by new web based technologies.

XML for Analysis

Based on these ideas, XML for Analysis was developed by Microsoft and Hyperion Solutions Corporation in April 2001. The specification defines a communication structure for an application programming interface (API) and aims to keep client programming independent of the mechanics of data transport while ensuring adequate information regarding data and proper handling of it. This is platform programming language and data source independent.

Simple Object Access Protocol (SOAP)

Another technology enabling XML use in data mining is SOAP specifically developed by Microsoft, IBM and Iona. SOAP standardises data access interaction between client applications and analytical providers (data mining and On Line Analytical Processing) over the Internet. SOAP can be described using WSDL (Web Service Description Language), which is the IDL (Interface Definition Language) for web service. WSDL is independent of SOAP, but needed to explain which SOAP messages can be exchanged. The means by which it is discovered is addressed later. Using the SOAP protocol, a server can retrieve information from a client across the web. In doing this, (1) the server side sends several SOAP requests, (2) processes the requests that it receives, (3) finds different patterns, and (4) creates profiles

based on appropriate limitations or performs appropriate analyses. Ease of use and the platform independence of this protocol are other important factors.

Explanation of processes of discovery

Universal Description Discovery and Integration (UDDI), developed by Microsoft, IBM and Ariba, uses the XML Schema Language to formally describe its data structures. UDDI is SOAP based and defines global interaction with the web service information repository. A web service is a self-describing, self-contained, modular unit of application logic that provides business functionality to other applications through an Internet connection. The UDDI specification enables businesses to quickly, easily and dynamically find each other and interact**.** It enables a business to describe it as well as to find and interact with businesses offering desired services. This internet facilitated discovery and interaction fosters new e-business partnerships. UDDI also simplifies the intergradation of disparate systems and allows market expansion, improved efficiency and reduced cost. Applications can access web services via ubiquitous web protocols and data formats, such as XML, without concern re web service implementation. Web services can be mixed and matched to execute a larger workflow or business transaction. UDDI Business Registry can be accessed using SOAP and a service registered in the UDDI Business Registry can expose to any type of service interface**.**

vTag Web Mining Server

A product that supports SOAP, WSDL and UDDI is vTag Web Mining Server. This product aims to monitor and mine the web and includes features (*Connotate Technologies: vTag*, 200), such as:

- Automatic extraction from HTML, PDF, spreadsheet, and other file formats and conversion to XML.
- Unlimited 'Information Agents' provide continuous monitoring, extraction and alerting.
- Scripting, password access, automated parameter entry, and multi-page aggregation.
- Seamless integration with other applications via Web Services, database delivery, and API programming interface.

Agent Repository filters extract and deliver information while instant Web Services create the web services. The information agents are accessed by SOAP and instant Web Services automatically generates WSDL and UDDI.(*Connotate Technologies: vTag*, 200)

Comments The combination of XML and data mining is possible with SOAP since this protocol enables data interaction on the web and therefore data collection. SOAP works optimally in collaboration with WSDL and UDDI. Some efforts have been made to implement these protocols, but in fact the full potential of these technologies has not yet been realised. There is much research in this area and new products are expected. IBM and Microsoft are developing database solutions (Xperanto, IBM/Yukon, Microsoft), which will support both data mining and XML. Since HTTP, XML and SOAP are platform independent, issues associated wth competing proprietary protocols should be resolved.

CONCLUSION AND FUTURE DIRECTIONS

With the growing importance of XML in document representation, new processing and integration technologies are being devised. The focus of this chapter, however, has been to describe, in general, the capability and benefits of data mining techniques in the XML domain and to conceptualize the XML mining process. This chapter attempts to show the improved knowledge discovery of both structure and content of XML documents with utilisation of data mining techniques.

This chapter explicitly expresses the representation of XML data and the broad categories of XML mining: XML structure mining and XML content mining. These categories are presented according to data mining tasks such as classification, clustering and association. This chapter then presents the process of knowledge discovery from XML documents summarising the three tasks of clustering, association mining and classification on structure or/and content of XML documents. The chapter further discussed the evolution of knowledge discovery where the current application of XML enables a simplified data mining process and makes the discovered patterns interchangeable among conforming data mining tools and other analytical applications. The chapter then introduces the protocols that support XML and data mining, making data mining possible across the web using XML.

XML data mining is a challenging and exciting field with further possibilities. Following are some of the areas identified for future development**:**

Integration of XML Mining The integration of XML, the database languages, such as SQL, and data mining techniques will increase the functionality of relational database products and XML products. It will provide more user friendly mining. The larger RDBMS and data warehouse companies have already expressed an interest in integrating data mining and XML data models into their database products.

Graphical user interface Full integration of data mining products with other application tools and the use of GUIs will enhance usability. To satisfy the range of data mining users (from naive to expert users), future work should include mining user graphs that is structural information of web usages**,** as well as visualization of mined data using systems such as WWWPal system (*WWWPAL*).

Multimedia XML data To perform web content mining, keyword information and content for each of the nodes is required. This information will allow the automatic development of a set of keywords to distinguish text document, multimedia document or other kinds of document based on the contained characteristics such as color, brightness and texture. Data mining is able to intelligently prepare data and allow types of information to be distinguished

Security and Privacy As data mining is applied to large semantic documents or XML documents, extraction of information should consider privacy and rights management of shared data. XML mining should have the authorization level to empower security to restrict only to appropriate users to discover classified information.

REFERENCES

Abiteboul, S., Buneman, P., & Suciu, D. (2000). *Data on the Web: From Relations to Semistructured Data and XML*: California: Morgan Kaumann.

Abiteboul, S., Quass, D., McHugh, J., Widom, J., and Weiner, J. (1997). The Lorel Query Language for Semistructured Data. *Journal of Digital Libraries, 1*(1), 68-88.

Adelberg, B. (1998). *NoDoSE: A tool for Semi-Automatically Extracting Structured and Semistructured Data from Text Documents.* Paper presented at the Proceedings of the ACM SIGMOD Conference on Management of Data, Seattle, USA.

Agrawal, R., & Srikant, R. (1996). *Mining Sequential Patterns: Generalizations and Performance Improvements.* Paper presented at the the 5th International Conference on Extending Database Technology (EDBT'96), France.

Bertino, E., Guerrini, G., & Mesiti, M. (2004). A Matching Algorithm for Measuring the Structural Similarity between an XML Document and a DTD and its applications. *Information Systems, 29*(1), 23-46.

Boag, S., Chamberlin, D., Fernández, M., Florescu, D., et al.*XQuery 1.0: An XML Query Language*. Retrieved September, 2005, from http://www.w3.org/TR/2005/WD-xquery-20050915/

Braga, D., Campi, A., Ceri, S., Klemettinen, M., et al. (2002). *A Tool for Extracting XML Association Rules.* Paper presented at the Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'02), USA.

Ceri, S., Comai, S., Damiani, E., Fraternali, P., Paraboschi, S., and Tanca, L. (1999). *XML-GL: A Graphical Language for Quering and Restructuring XML Documents.* Paper presented at the Proc. 8th International WWW Conference, Toronto, Canada.

Chi, Y., Nijssen, S., & Muntz, R. (2005). Frequent Subtree Mining - An Overview. *Fundamenta Informatiace Special Issue on Graph and Tree Mining, 66*(1-2), 161-198.

Connotate Technologies: vTag. (200). 2006, from http://www.connotate.com/csp.asp

Dalamagas, T., Cheng, T., Winkel, K., & Sellis, T. K. (2004). *Clustering XML documents by Structure*. Paper presented at the SETN.

Do, H. H., & Rahm, E. (2002). *COMA - A System for Flexible Combination of Schema Matching Approaches.* Paper presented at the 28th VLDB, Hong Kong, China.

Doan, A., Domingos, R., & Halevy, A. Y. (2001). *Reconciling schemas of disparate sources: a machine-learning approach.* Paper presented at the ACM SIGMOD, Santa Barbara, California, United States.

Edmonds, A. (2005). *XML Miner & Metarule White Paper*. Retrieved January 14, 2005, from www.scientio.com/resources/

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. (1995). From Data Mining to Knowledge Discovery: An Overview. In U. M. Fayyad, Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R (Ed.), *Advances in Knowledge Discovery and Data Mining* (pp. 1-34): AAAI Press.

Feng, L., Chang, E., & Dillon, T. (2002). A Semantic Network-Based Design Methodology for XML Documents. *ACM Transactions of Information Systems (TOIS), 20*(4), 390 - 421.

Fernandez, M., Buneman, P., and Suciu, D. (2000). (2000). UNQL: A Query Language and Algebra for Semistructured Data based on Structural Recursion. *VLDB JOURNAL: Very Large Data Bases, 9*(1), 76-110.

Flesca, S., Manco, G., Masciari, E., Pontieri, L., et al. (2005). Fast Detection of XML Structural Similarities. *IEEE Transaction on Knowledge and Data Engineering, 7*(2), 160-175.

Garofalakis, M., Rastogi, R., Seshadri, S., and Shim, K. (1999). *Data Mining and the Web: Past, Present and Future.* Paper presented at the The second international workshop on web information and data management, Kansas City, USA.

Garofalakis, M. N., Gionis, A., Rastogi, R., Seshadri, S., & Shim, K. (2000). *XTRACT: A System for Extracting Document Type Descriptors from XML Documents.* Paper presented at the Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Texas, USA.

Guardalben, G. (2004). *Integrating XML and Relational Database Technologies: A Position Paper*. Retrieved May 1st, 2005, from http://www.hitsw.com/products_services/whitepapers/integrating_xml_rdb/integrating_xml_white_paper.pdf

Han, J., & Kamber, M. (2001). *Data Mining: Concepts and Techiques.* San Diego, USA: Morgan Kaufmann.

Huang, Z. (1997). *A fast clustering algorithm to cluster very large categorical data sets in data mining.* Paper presented at the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data Clustering: A Review. *ACM Computing Surveys (CSUR), 31*(3), 264-323.

Jeong, E., & Hsu, C.-N. (2001). *Induction of integrated view for XML data with heterogeneous DTDs.* Paper presented at the 10th International Conference on Information and Knowledge Management, Atlanta, Georgia, USA.

Kay, M. (2000). *SAXON DTD Generator - A Tool to Generate XML DTDs*, January, 2006, from http://home.iclweb.com/ic2/mhkay/dtdgen.html

Kotasek, P., and Zendulka, J. (2000). *An XML Framework Proposal for Knowledge Discovery in Database.* Paper presented at the Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases, Workshop Proceedings Knowledge Management: Theory and Applications, Lyon, French.

Kotásek, P., & Zendulka, J. (2002). *Describing the Data Mining Process with DMSL.* Paper presented at the ADBIS 2002.

Kurgan, L., Swiercz, W., & Cios, K. (2002). *Semantic Mapping of XML Tags using Inductive Machine Learning.* Paper presented at the International Conference on Machine Learning and Applications 2002 (ICMLA).

Kutty, S., Nayak, R., & Li, Y. (2007). *PCITMiner- Prefix-based Closed Induced Tree Miner for finding closed induced frequent subtrees.* Paper presented at the the Sixth Australasian Data Mining Conference (AusDM 2007), Gold Coast, Australia.

Lee, J. W., & Park, S. S. (2004). *Finding Maximal Similar Paths Between XML Documents Using Sequential Patterns.* Paper presented at the ADVIS, Izmir, Turkey.

Lee, L. M., Yang, L. H., Hsu, W., & Yang, X. (2002). *XClust: Clustering XML Schemas for Effective Integration.* Paper presented at the 11th ACM International Conference on Information and Knowledge Management (CIKM'02), Virginia.

Leung, H.-p., Chung, F.-l., & Chan, S. C.-f. (2005). On the use of hierarchical information in sequential mining-based XML document similarity computation. *Knowledge and Information Systems, 7*(4), 476-498.

Madhavan, J., Bernstein, P. A., & Rahm, E. (2001). *Generic Schema Matching with Cupid.* Paper presented at the 27th VLDB, Roma, Italy.
Melnik, S., Garcia-Molina, H., & Rahm, E. (2002). *Similarity Flooding: A Versatile Graph Matching Algorithm.* Paper presented at the ICDE.
Moh, C.-H., E.-P. Lim, et al. (2000). *DTD-Miner: a tool for mining DTD from XML documents.* Paper presented at the Proceedings of the Second International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems.
Myaeng, S. H., Jang, D.H., Kim, M.S., & Zhoo, Z.C. (1998). *A Flexible Model for Retrieval of SGML Documents.* Paper presented at the Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia.
Nayak, R., & Iryadi, W. (2007). XML schema clustering with semantic and hierarchical similarity measures. *Knowledge-Based Systems, 20*(4), 336-349.
Nayak, R., Witt, R., & Tonev, A. (2002). *Data Mining and XML documents.* Paper presented at the The 2002 International Workshop on the Web and Database (WebDB 2002).
Nayak, R., & Xia, F. B. (2004). *Automatic integration of heterogeneous XML-schemas.* Paper presented at the Proceedings of the International Conferences on Information Integration and Web-based Applications & Services.
Nayak, R., & Xu, S. (2006). *XCLS: A Fast and Effective Clustering Algorithm for Heterogenous XML Documents.* Paper presented at the the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Singapore.
Nayak, R., & Zaki, M. (Eds.). (2006). *Knowledge Discovery from XML documents: PAKDD 2006 Workshop Proceedings* (Vol. 3915): Springer-Verlag Heidelberg.
Nestorov, S., Ullman, J., Weiner, J., and Chawathe, S. (1999). *Representative objects: Concise Representation of Semistructured, Hierarchical data.* Paper presented at the IEEE Proc on Management of Data, Seattle, USA.
Paik, J., Won, D., Fotouh, i. F., & Kim, U. (2005). *ExiT-B: A new approach for extracting maximal frequent subtrees from XML data.* Paper presented at the IDEAL.
Pardede, E., Rahayu, J.W., and Taniar, D. (2006). Object-Relational Complex Structures for XML Storage. *Information and Software Technology Journal, 48*(6), 370-384.
Polyzotis, N., Garofalakis, M., & Ioannidis, Y. (2004). *Approximate XML query answers.* Paper presented at the International Conference on Management of Data.
Punin, J., Krishnamoorthy, M., & Zaki, M. (2001). *LOGML - Log Markup Language for Web Usage Mining.* Paper presented at the Proceedings of the WEBKDD Workshop 2001: Mining Log Data Across All Customer TouchPoints (with SIGKDD01), San Francisco, USA.
Shanmugasundaram, J., Shekita, E. J., Kiernan, J., Krishnamurthy, R, Viglas, S., Naughton, J. F., and Tatarinov, I. (2001). A General Technique for Querying XML Documents using a Relational Database System. *SIGMOD Record, 30*(3), 20-26.
Theobald, M., Schenkel, R., & Weikum, G. (2003). *Exploiting structure, annotation, and ontological knowledge for automatic classification of XML data.* Paper presented at the International Workshop on the Web and Databases (WebDB), San Diego, California.
Vianu, V. (2001). *A Web Odyssey: from Codd to XML.* Paper presented at the Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems.
Wan, J. W. W. D., G. (2004). Mining Association rules from XML data mining uqery. *Reearch and practice in Information Technology, 32*, 169-174.
Wang, Q., Yu, Xu. J., and Wong, K. (2000). *Approximate Graph Scheme Extraction for Semi-structured data.* Paper presented at the Proc of 7th International Conference on Extending Database Technology (EDBT-2000), Konstanz.
Wettschereck, D., and Muller, S. (2001). *Exchanging Data Mining Models with the Predictive Modeling Markup Language.* Paper presented at the Integration aspects of Data Mining, Decision Support and Meta-Learning IDDM-2001. A workshop held at the European Conference on Machine Learning (ECML-2001) and the Conference on Principles and Practice of Knowledge Discovery and Data Mining (PKDD-2001), Freiburg, Germany.
WWWPAL. 2006, from http://www.rpi.edu/~prestn2/summaryNP.html
Yergeau, F., Bray, T., Paoli, J., Sperberg-McQueen, C. M., et al. (2004). *Extensible Markup Language (XML) 1.0 (Third Edition) W3C Recommendation*. Retrieved February, 2004, from http://www.w3.org/TR/2004/REC-XML-20040204/
Zaki, M. J. (2002). *Efficiently mining Frequent trees in a forest.* Paper presented at the ACM SIGKDD Conference on knowledge discovery in databases.
Zaki, M. J., & Aggarwal, C. C. (2003). *XRules: An Effective Structural Classifier for XML Data.* Paper presented at the SIGKDD.

Zhang, K., & Shasha, D. (1989). Simple Fast Algorithms for the Editing Distance Between Trees and Related Problems. *SIAM Journal Computing, 18*(6), 1245-1262.

Zhang, M., & Yao, J. T. (2004). *The XML Algebra for Data Mining.* Paper presented at the Proceedings of Data Mining and Knowledge Discovery: Theory, Tools, and Technology, Orlando, USA.

Zhao, Q., Chen, L., Bhowmick, S. S., & Madria, S. (2007). XML structural delta mining: Issues and challenges. *Data & Knowledge Engineering, In Press, Corrected Proof.*